

## Program e2m

### A Computer Program for Preliminary Earth-to-Mars Mission Design

This document is the user's manual for a Fortran computer program (e2m.f) that can be used to design ballistic interplanetary missions from Earth park orbit to B-plane encounter at Mars. The software assumes that interplanetary injection occurs *impulsively* from a circular Earth park orbit. The B-plane coordinates are expressed in a Mars-centered (areocentric) mean equator and IAU node of epoch coordinate system. These B-plane targets are enforced via a user-defined periapsis radius and orbital inclination of the arrival hyperbola.

The first part of this computer program solves for the minimum delta-v using a patched-conic, two-body Lambert solution for the transfer trajectory from Earth to Mars. The second part implements a simple *shooting* method that attempts to minimize an impulsive trajectory correction maneuver (TCM) located at the Earth's sphere-of-influence (SOI) while numerically integrating the spacecraft's heliocentric equations of motion and targeting to components of the B-plane relative to Mars. The spacecraft motion within the Earth's SOI includes the Earth's  $J_2$  oblate gravity effect and the point-mass perturbation of the sun and moon. The heliocentric equations of motion include the point-mass gravity of the sun and the first seven planets of the solar system.

The user can select one of the following delta-v optimization options for the two-body solution of the interplanetary transfer trajectory:

- minimize launch delta-v
- minimize arrival delta-v
- minimize total delta-v

The major computational steps implemented in this software are as follows:

- solve the two-body, patched-conic interplanetary Lambert problem for the energy  $C_3$ , declination (DLA) and asymptote (RLA) of the outgoing hyperbola
- compute the orbital elements of the geocentric launch hyperbola and the components of the interplanetary injection delta-v vector
- perform geocentric orbit propagation from perigee of the geocentric launch hyperbola to the Earth's sphere-of-influence (SOI)
- perform an n-body heliocentric orbit propagation from the Earth's SOI to the B-plane at Mars encounter
- target to the user-defined B-plane coordinates by minimizing a heliocentric delta-v vector or trajectory correction maneuver (TCM) at the Earth's sphere-of-influence

This computer program uses the SNOPT nonlinear programming algorithm to solve both the patched-conic and numerically integrated trajectory optimization problems. With the appropriate coordinate transformations, the software can be easily modified for any combination of departure and arrival planets within our solar system.

## Input data file

This section describes a typical input data file for the software. In the following discussion the actual input file contents are in *courier* font and all explanations are in *times italic* font.

Each data item within an input file is preceded by one or more lines of *annotation* text. Do not delete any of these annotation lines or increase or decrease the number of lines reserved for each comment. However, you may change them to reflect your own explanation. The annotation line also includes the correct units and when appropriate, the valid range of the input.

*The first six lines of any input file are reserved for user comments. These lines are ignored by the software. However the input file must begin with six and only six initial text lines.*

```
*****  
** earth-to-mars interplanetary  
** trajectory optimization  
** Mars '03 example - mars03.in  
** April 15, 2005  
*****
```

*The first input is an integer that defines the type of patched-conic trajectory optimization.*

```
*****  
* simulation type *  
*****  
1 = minimize launch delta-v  
2 = minimize arrival delta-v  
3 = minimize total delta-v  
-----  
1
```

*The next input defines an initial guess for the launch calendar date. Please be sure to include all digits of the calendar year.*

```
launch calendar date initial guess (month, day, year)  
6,1,2003
```

*These two numbers define the lower and upper search interval for the launch calendar date.*

```
launch date search boundary (days)  
-30, +30
```

*The next input defines an initial guess for the arrival calendar date.*

```
arrival calendar date initial guess (month, day, year)  
12,1,2003
```

*The software allows the user to specify an initial guess for the launch and arrival calendar dates and a search interval. For any guess for launch time  $t_L$  and user-defined search interval  $\Delta t$ , the launch time  $t$  is constrained as follows:*

$$t_L - \Delta t \leq t \leq t_L + \Delta t$$

*Likewise, for any guess for arrival time  $t_A$  and user-defined search interval, the arrival time  $t$  is constrained as follows:*

$$t_A - \Delta t \leq t \leq t_A + \Delta t$$

*For fixed launch and/or arrival times, the search interval should be set to 0.*

*These two numbers define the lower and upper search interval for the arrival calendar date.*

```
arrival date search boundary (days)
-30, +30
```

*The next set of inputs defines the characteristics of the launch hyperbola.*

```
*****
* geocentric phase modeling
*****

perigee altitude of launch hyperbola (kilometers)
185.2d0

launch azimuth (degrees)
93.0d0

launch site latitude (degrees)
28.5d0
```

*The next three integer inputs define the types of perturbations to include during the numerical integration of the spacecraft's geocentric motion. The first option will include the effect of  $J_2$  in the equations of motion, and options 2 and 3 will include the point mass gravity of the moon and sun.*

```
*****
trajectory perturbations
*****

include Earth j2 gravity perturbations (1 = yes, 0 = no)
1

include solar perturbations (1 = yes, 0 = no)
1

include lunar perturbations (1 = yes, 0 = no)
1
```

*These next two inputs define the radius of closest approach and the orbital inclination of the encounter hyperbola at Mars.*

```
*****
* encounter planet targeting
*****

radius of closest approach (kilometers)
5000.0d0

orbital inclination (degrees)
60.0d0
```

*The next two inputs are algorithm control parameters. The first input is the truncation error tolerance for the Runge-Kutta-Fehlberg(RKF7(8)) integrator and determines how well the equations of orbital motion are solved. The second input is the root-finding tolerance and it determines how accurately close approach to Mars is predicted.*

```
*****
algorithm control parameters
```

```

*****
truncation error tolerance
1.0d-10

root-finding tolerance
1.0d-6

```

*The final two inputs specify the name of the solution disk file and the time step at which the data is created and written.*

```

*****
output file characteristics
*****

name of output data file
mars03.csv

print step size (days)
0.25

```

## Program example

The following is the solution created with this Computer program for this example. The output is organized in the following major sections:

- First Pass
  1. two body Lambert solution
  2. departure hyperbola orbital elements
  3. time and conditions at Earth SOI
- Targeting Pass
  1. time and conditions at Mars closest approach
  2. TCM characteristics

*The first output section summarizes the two-body Lambert solution. The solution is provided in the heliocentric, Earth mean equator and equinox of J2000 (EME2000) coordinate system. The time scale is Barycentric Dynamical Time (TDB)*

```

=====
two-body Lambert solution
=====

minimize launch delta-v

departure heliocentric delta-v vector and magnitude
(Earth mean equator and equinox of J2000)
-----

x-component of delta-v      2895.91228327510      meters/second
y-component of delta-v      -530.400048454818      meters/second
z-component of delta-v      -345.700227274109      meters/second

delta-v magnitude          2964.31118659324      meters/second

arrival heliocentric delta-v vector and magnitude
(Earth mean equator and equinox of J2000)
-----

x-component of delta-v      -2063.01097067142      meters/second
y-component of delta-v       1164.27062001093      meters/second

```

z-component of delta-v      1311.96074499657      meters/second  
delta-v magnitude            2707.91088076089      meters/second

heliocentric coordinates of the Earth at departure  
(Earth mean equator and equinox of J2000)  
-----

calendar date                June            5, 2003  
TDB time                      14:47:19.195  
TDB Julian date               2452796.11619439

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.149651463878D+09	0.162373945919D-01	0.234390546206D+02	0.102452247269D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.723854488397D-03	0.152047949297D+03	0.254500196566D+03	0.526252600676D+06
rx (km)	ry (km)	rz (km)	rmag (km)
-.405616863560D+08	-.134199732713D+09	-.581818246266D+08	0.151789141170D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.282279741011D+02	-.739769564570D+01	-.320741192653D+01	0.293569738561D+02

heliocentric coordinates of the spacecraft after the first impulse  
(Earth mean equator and equinox of J2000)  
-----

calendar date                June            5, 2003  
TDB time                      14:47:19.195  
TDB Julian date               2452796.11619439

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.188387162818D+09	0.194277237656D+00	0.234900282797D+02	0.253491131248D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.455892127356D+00	0.591536035565D+00	0.254082667283D+03	0.743275399874D+06
rx (km)	ry (km)	rz (km)	rmag (km)
-.405616863560D+08	-.134199732713D+09	-.581818246266D+08	0.151789141170D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.311238863844D+02	-.792809569416D+01	-.355311215381D+01	0.323137062403D+02

heliocentric coordinates of the spacecraft prior to the second impulse  
(Earth mean equator and equinox of J2000)  
-----

calendar date                December 24, 2003  
TDB time                      15:24:20.195  
TDB Julian date               2452998.14190041

sma (km)	eccentricity	inclination (deg)	argper (deg)
0.188387162818D+09	0.194277237656D+00	0.234900282797D+02	0.253491131248D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.455892127356D+00	0.152910265597D+03	0.464013968448D+02	0.743275399874D+06
rx (km)	ry (km)	rz (km)	rmag (km)
0.149989640878D+09	0.146777505371D+09	0.632696138262D+08	0.219188440588D+09
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
-.146794819119D+02	0.156262553589D+02	0.684180295615D+01	0.225050508427D+02

heliocentric coordinates of the spacecraft after the second impulse  
 (Earth mean equator and equinox of J2000)

```
-----
calendar date          December 24, 2003
TDB time              15:24:20.195
TDB Julian date       2452998.14190041

   sma (km)            eccentricity      inclination (deg)      argper (deg)
0.227939306995D+09   0.935418896809D-01  0.246772249523D+02   0.332979237038D+03

   raan (deg)          true anomaly (deg)      arglat (deg)          period (min)
0.337165832669D+01   0.707599701626D+02  0.437392072004D+02   0.989239925943D+06

   rx (km)             ry (km)                   rz (km)               rmag (km)
0.149989640878D+09   0.146777505371D+09   0.632696138262D+08   0.219188440588D+09

   vx (kps)            vy (kps)                   vz (kps)              vmag (kps)
-.167424928826D+02   0.167905259789D+02   0.815376370115D+01   0.250742236782D+02
```

heliocentric coordinates of Mars at arrival  
 (Earth mean equator and equinox of J2000)

```
-----
calendar date          June      5, 2003
TDB time              14:47:19.195
TDB Julian date       2452796.11619439

   sma (km)            eccentricity      inclination (deg)      argper (deg)
0.227939306995D+09   0.935418896818D-01  0.246772249523D+02   0.332979237038D+03

   raan (deg)          true anomaly (deg)      arglat (deg)          period (min)
0.337165832666D+01   0.707599701624D+02  0.437392071999D+02   0.989239925943D+06

   rx (km)             ry (km)                   rz (km)               rmag (km)
0.149989640879D+09   0.146777505370D+09   0.632696138256D+08   0.219188440588D+09

   vx (kps)            vy (kps)                   vz (kps)              vmag (kps)
-.167424928825D+02   0.167905259790D+02   0.815376370122D+01   0.250742236783D+02
```

*The following output summarizes the characteristics of the initial circular park orbit and the departure hyperbola.*

-----  
 park orbit and departure hyperbola characteristics  
 (Earth mean equator and equinox of J2000)

-----  
 park orbit

```
-----
calendar date          June      5, 2003
TDB time              14:47:19.195
TDB Julian date       2452796.11619439

   sma (km)            eccentricity      inclination (deg)      argper (deg)
0.656334000000D+04   0.157151442999D-15  0.286442848562D+02   0.000000000000D+00

   raan (deg)          true anomaly (deg)      arglat (deg)          period (min)
0.203490691526D+01   0.195040136990D+03  0.195040136990D+03   0.881956335064D+02

   rx (km)             ry (km)                   rz (km)               rmag (km)
-.628143605937D+04   -.171884062736D+04   -.816443436710D+03   0.656334000000D+04

   vx (kps)            vy (kps)                   vz (kps)              vmag (kps)
```

0.225551439551D+01    -.652900749392D+01    -.360777732677D+01    0.779303158492D+01

hyperbola  
-----

c3                    8.78714081096184            km\*\*2/sec\*\*2  
v-infinity            2964.31118659324            meters/second  
declination           -6.69711691578417            degrees  
right ascension       349.621042641743            degrees  
inclination            28.6442848562298            degrees  
perigee altitude      185.200000000000            kilometers  
perigee radius        6563.340000000000            kilometers  
launch azimuth        93.000000000000            degrees  
launch latitude       28.500000000000            degrees

calendar date            June            5, 2003  
TDB time                14:47:19.195  
TDB Julian date          2452796.11619439

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.453617906069D+05	0.114468873279D+01	0.286442848562D+02	0.195040136990D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.203490691525D+01	0.000000000000D+00	0.195040136990D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
-.628143605937D+04	-.171884062736D+04	-.816443436710D+03	0.656334000000D+04
vx (kps)	vy (kps)	vz (kps)	vmag (kps)
0.330314575902D+01	-.956157205515D+01	-.528350180344D+01	0.114127044726D+02

departure delta-v vector and magnitude

delta-vx	1047.63136350357	meters/seconds
delta-vy	-3032.56456122821	meters/seconds
delta-vz	-1675.72447667587	meters/seconds
delta-v magnitude	3619.67288764388	meters/seconds

*This section of the program output summarizes the flight conditions at the Earth's sphere-of-influence.*

spacecraft geocentric coordinates at the Earth SOI prior to the TCM  
(Earth mean equator and equinox of J2000)  
-----

calendar date            June            8, 2003  
TDB time                18:22:31.155  
TDB Julian date          2452799.26563837

sma (km)	eccentricity	inclination (deg)	argper (deg)
-.456136808308D+05	0.114313621030D+01	0.284940995714D+02	0.194983224362D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.205409028814D+01	0.149491459967D+03	0.344474684329D+03	0.000000000000D+00
rx (km)	ry (km)	rz (km)	rmag (km)
0.898475527675D+06	-.185513072562D+06	-.118117001646D+06	0.925000000000D+06
vx (kps)	vy (kps)	vz (kps)	vmag (kps)

0.302773557360D+01    -.552100755001D+00    -.358408779637D+00    0.309846006923D+01

spacecraft heliocentric coordinates at the Earth SOI prior to the TCM  
(Earth mean equator and equinox of J2000)

-----  
calendar date            June            8, 2003

TDB time                18:22:31.155

TDB Julian date         2452799.26563837

      sma (km)            eccentricity        inclination (deg)        argper (deg)  
0.190724813493D+09    0.204052839793D+00    0.234926771747D+02    0.253488392966D+03

      raan (deg)          true anomaly (deg)    arglat (deg)            period (min)  
0.463119088547D+00    0.394999931898D+01    0.257438392285D+03    0.757152915961D+06

      rx (km)             ry (km)             rz (km)             rmag (km)  
-.319297502743D+08    -.136208380815D+09    -.590902781310D+08    0.151868011532D+09

      vx (kps)            vy (kps)            vz (kps)            vmag (kps)  
0.316261387715D+02    -.655234197930D+01    -.295906278828D+01    0.324330370393D+02

spacecraft heliocentric coordinates at the Earth SOI prior to the TCM  
(Earth mean ecliptic and equinox of J2000)

-----  
calendar date            June            8, 2003

TDB time                18:22:31.155

TDB Julian date         2452799.26563837

      sma (km)            eccentricity        inclination (deg)        argper (deg)  
0.190724813493D+09    0.204052839793D+00    0.192000939739D+00    0.179847963055D+03

      raan (deg)          true anomaly (deg)    arglat (deg)            period (min)  
0.740652731537D+02    0.394999931906D+01    0.183797962374D+03    0.757152915962D+06

      rx (km)             ry (km)             rz (km)             rmag (km)  
-.319296848989D+08    -.148473522934D+09    -.337097979317D+05    0.151868011532D+09

      vx (kps)            vy (kps)            vz (kps)            vmag (kps)  
0.316261419164D+02    -.718868986334D+01    -.108522465326D+00    0.324330370393D+02

*This section of the program output summarizes the TCM maneuver characteristics and the flight conditions at closest approach to Mars. It includes the B-plane coordinates.*

heliocentric TCM delta-v vector and magnitude  
(Earth mean equator and equinox of J2000)

-----

deltav-x                2.86345926835030        meters/second  
deltav-y                19.6988182704806        meters/second  
deltav-z                -2.66005685633381        meters/second

delta-v                 20.0827971835064        meters/second

transfer time            198.323548421264        days

spacecraft heliocentric coordinates at the Earth SOI after the TCM  
(Earth mean equator and equinox of J2000)

-----

calendar date            June            8, 2003

TDB time                18:22:31.155

```

TDB Julian date          2452799.26563837

      sma (au)            eccentricity      inclination (deg)      argper (deg)
0.127480509543D+01     0.203954808611D+00   0.234966345853D+02   0.253625880369D+03

      raan (deg)          true anomaly (deg)    arglat (deg)          period (days)
0.507594179943D+00     0.377172393124D+01   0.257397604300D+03   0.525731637632D+03

      rx (km)             ry (km)                rz (km)                rmag (km)
-.319297502743D+08     -.136208380815D+09   -.590902781310D+08   0.151868011532D+09

      vx (kps)            vy (kps)                vz (kps)                vmag (kps)
0.316290022308D+02     -.653264316103D+01   -.296172284513D+01   0.324320984674D+02

```

time and conditions at Mars closest approach  
(areocentric mean equator and IAU node of epoch)  
-----

calendar date December 24, 2003

TDB time 02:08:25.739

TDB Julian date 2452997.58918679

```

      sma (km)            eccentricity      inclination (deg)      argper (deg)
-.584680813740D+04     0.185516669334D+01   0.600000403908D+02   0.113982634156D+03

      raan (deg)          true anomaly (deg)    arglat (deg)          period (min)
0.105661018007D+03     0.241929148834D-05   0.113982636575D+03   0.000000000000D+00

      rx (km)             ry (km)                rz (km)                rmag (km)
-.165075937539D+04     -.257344968217D+04   0.395629953777D+04   0.4999999558146D+04

      vx (kps)            vy (kps)                vz (kps)                vmag (kps)
0.218745295892D+01     -.407935903568D+01   -.174078478880D+01   0.494534652683D+01

```

b-plane coordinates at closest approach  
(areocentric mean equator and IAU node of epoch)  
-----

```

b-magnitude            9136.08151889761      kilometers
b dot r                -7889.37005619595
b dot t                4607.14940460381
theta                  300.283613961100      degrees
vinf                   2.70648972776838      km/sec
r-periapsis           4999.99558146210      kilometers
decl-asy               7.47130685486324      degrees
rasc-asy              281.318691800287      degrees

fpa                    1.571953386226682E-006 degrees

```

*From the results of this simulation, we can see that the n-body effects and B-plane targeting require a 20 meters/second trajectory correction maneuver at the Earth's sphere-of-influence.*

The e2m software suite also includes a MATLAB script called `marsplot.m` that can be used create a graphics display of the interplanetary and encounter trajectories. The interactive graphic features of MATLAB will allow the user to rotate and “zoom” the displays in and out. These capabilities allow the user to interactively find the “best” viewpoint as well as verify basic orbital geometry of the heliocentric and areocentric trajectories.

The following is a heliocentric, ecliptic view of the transfer and planetary orbits in an ecliptic coordinate system. The x-axis of this system is red, the y-axis green and the z-axis is blue.



## Technical discussion

### *Solving the two body Lambert problem*

Lambert's problem is concerned with the determination of an orbit that passes between two positions within a specified time-of-flight. This classic astrodynamic problem is also known as the orbital two-point boundary value problem (TPBVP).

The time to traverse a trajectory depends only upon the length of the semimajor axis  $a$  of the transfer trajectory, the sum  $r_i + r_f$  of the distances of the initial and final positions relative to a central body, and the length  $c$  of the chord joining these two positions. This relationship can be stated as follows:

$$tof = tof(r_i + r_f, c, a)$$

From the following form of Kepler's equation

$$t - t_0 = \sqrt{\frac{a^3}{\mu}} (E - e \sin E)$$

we can write

$$t = \sqrt{\frac{a^3}{\mu}} [E - E_0 - e(\sin E - \sin E_0)]$$

where  $E$  is the eccentric anomaly associated with radius  $r$ ,  $E_0$  is the eccentric anomaly at  $r_0$ , and  $t = 0$  when  $r = r_0$ .

At this point we need to introduce the following trigonometric sum and difference identities:

$$\sin \alpha - \sin \beta = 2 \sin \frac{\alpha - \beta}{2} \cos \frac{\alpha + \beta}{2}$$

$$\cos \alpha - \cos \beta = -2 \sin \frac{\alpha - \beta}{2} \sin \frac{\alpha + \beta}{2}$$

$$\cos \alpha + \cos \beta = 2 \cos \frac{\alpha - \beta}{2} \cos \frac{\alpha + \beta}{2}$$

If we let  $E = \alpha$  and  $E_0 = \beta$  and substitute the first trig identity into the second equation above, we have the following equation:

$$t = \sqrt{\frac{a^3}{\mu}} \left\{ E - E_0 - 2 \sin \frac{E - E_0}{2} \left( e \cos \frac{E + E_0}{2} \right) \right\}$$

With the two substitutions given by

$$e \cos \frac{E + E_0}{2} = \cos \frac{\alpha + \beta}{2}$$

$$\sin \frac{E - E_0}{2} = \sin \frac{\alpha - \beta}{2}$$

the time equation becomes

$$t = \sqrt{\frac{a^3}{\mu}} \left\{ (\alpha - \beta) - 2 \sin \frac{\alpha - \beta}{2} \cos \frac{\alpha + \beta}{2} \right\}$$

From the elliptic relationships given by

$$r = a(1 - e \cos E)$$

$$x = a(\cos E - e)$$

$$y = a \sin E \sqrt{1 - e^2}$$

and some more manipulation, we have the following equations:

$$\cos \alpha = \left( 1 - \frac{r + r_0}{2a} \right) - \frac{c}{2a} = 1 - \frac{r + r_0 + c}{2a} = 1 - \frac{s}{a}$$

$$\sin \beta = \left( 1 - \frac{r + r_0}{2a} \right) + \frac{c}{2a} = 1 - \frac{r + r_0 - c}{2a} = 1 - \frac{s - c}{a}$$

This part of the derivation makes use of the following three relationships:

$$\cos \frac{\alpha - \beta}{2} \cos \frac{\alpha + \beta}{2} = 1 - \frac{r + r_0}{2}$$

$$\sin \frac{\alpha - \beta}{2} \sin \frac{\alpha + \beta}{2} = \sin \frac{E - E_0}{2} \sqrt{1 - \left( e \cos \frac{E + E_0}{2} \right)^2}$$

$$\left( \sin \frac{\alpha - \beta}{2} \sin \frac{\alpha + \beta}{2} \right)^2 = \left( \frac{x - x_0}{2a} \right)^2 + \left( \frac{y - y_0}{2a} \right)^2 = \left( \frac{c}{2a} \right)^2$$

With the use of the half angle formulas given by

$$\sin \frac{\alpha}{2} = \sqrt{\frac{s}{2a}} \quad \sin \frac{\beta}{2} = \sqrt{\frac{s - c}{2a}}$$

and several additional substitutions, we have the time-of-flight form of Lambert's theorem

$$t = \sqrt{\frac{a^3}{\mu}} [(\alpha - \beta) - (\sin \alpha - \sin \beta)]$$

A discussion about the angles  $\alpha$  and  $\beta$  can be found in “Geometrical Interpretation of the Angles  $\alpha$  and  $\beta$  in Lambert’s Problem” by J. E. Prussing, *AIAA Journal of Guidance and Control*, Volume 2, Number 5, Sept.-Oct. 1979, pages 442-443.

The algorithm used in this computer program is based on the method described in “A Procedure for the Solution of Lambert’s Orbital Boundary-Value Problem” by R. H. Gooding, *Celestial Mechanics and Dynamical Astronomy* **48**: 145-165, 1990. This iterative solution is valid for elliptic, parabolic and hyperbolic transfer orbits which may be either posigrade or retrograde, and involve one or more revolutions about the central body.

### *Designing the launch hyperbola*

This section describes the algorithm used to determine the Earth-centered-inertial (ECI) state vector of a departure hyperbola for interplanetary missions. In the discussion that follows, interplanetary injection is assumed to occur *impulsively* at perigee of the departure hyperbola.

The departure trajectory for interplanetary missions is specified by the orbital energy  $C_3$ , and the right ascension  $\alpha_\infty$  and declination  $\delta_\infty$  of the outgoing asymptote. The perigee radius of the departure hyperbola is specified and the orbital inclination is computed from the user-defined launch azimuth  $\Sigma_L$  and launch site geocentric latitude  $\phi_L$  from the equation  $i = \cos^{-1}(\cos \phi_L \sin \Sigma_L)$ .

The algorithm used to design the departure hyperbola only works for geocentric orbit inclinations that satisfy the following constraint

$$|i| > |\delta_\infty|$$

If this inequality is not satisfied, the software will print the following error message

```
park orbit error!!
|inclination| must be > |asymptote declination|
```

The code will also print the inclination of the park orbit, the declination of the launch hyperbola and stop. The user can then change either the azimuth or launch site latitude to satisfy this constraint and restart the program.

A unit vector in the direction of the departure asymptote is given by

$$\hat{\mathbf{S}} = \begin{Bmatrix} \cos \delta_\infty \cos \alpha_\infty \\ \cos \delta_\infty \sin \alpha_\infty \\ \sin \delta_\infty \end{Bmatrix}$$

where

$\alpha_\infty$  = right ascension of departure asymptote

$\delta_\infty$  = declination of departure asymptote

The T-axis direction of the B-plane coordinate system is determined from the following vector cross product:

$$\hat{\mathbf{T}} = \hat{\mathbf{S}} \times \hat{\mathbf{u}}_z$$

where  $\hat{\mathbf{u}}_z = [0 \ 0 \ 1]^T$  is a unit vector perpendicular to the Earth's equator.

The following cross product operation completes the B-plane coordinate system.

$$\hat{\mathbf{R}} = \hat{\mathbf{S}} \times \hat{\mathbf{T}}$$

The B-plane angle is determined from the orbital inclination of the departure hyperbola  $i$  and the declination of the outgoing asymptote according to

$$\cos \theta = \frac{\cos i}{\cos \delta_\infty}$$

The unit angular momentum vector of the departure hyperbola is given by

$$\hat{\mathbf{h}} = \hat{\mathbf{T}} \sin \theta - \hat{\mathbf{R}} \cos \theta$$

The sine and cosine of the true anomaly at infinity are given by the next two equations

$$\cos \theta_\infty = -\frac{\mu}{r_p V_\infty^2 + \mu}$$

$$\sin \theta_\infty = \sqrt{1 - \cos^2 \theta_\infty}$$

where  $V_\infty = \sqrt{C_3} = V_L - V_p$  is the spacecraft's velocity at infinity,  $V_L$  is the heliocentric departure velocity determined from the Lambert solution,  $V_p$  is the heliocentric velocity of the departure planet, and  $r_p$  is the user-specified perigee radius of the departure hyperbola.

A unit vector in the direction of perigee of the departure hyperbola is determined from

$$\hat{\mathbf{r}}_p = \hat{\mathbf{S}} \cos \theta_\infty - (\hat{\mathbf{h}} \times \hat{\mathbf{S}}) \sin \theta_\infty$$

The ECI position vector at perigee is

$$\mathbf{r}_p = r_p \hat{\mathbf{r}}_p$$

The scalar magnitude of the perigee velocity can be determined from

$$V_p = \sqrt{\frac{2\mu}{r_p} + V_\infty^2}$$

A unit vector aligned with the velocity vector at perigee is

$$\hat{\mathbf{v}}_p = \hat{\mathbf{h}} \times \hat{\mathbf{r}}_p$$

The ECI velocity vector at perigee of the departure hyperbola is given by

$$\mathbf{v}_p = V_p \hat{\mathbf{v}}_p$$

Finally, the classical orbital elements of the departure hyperbola can be determined from the position and velocity vectors at perigee. The injection delta-v vector and magnitude can be determined from the velocity difference between the park orbit and departure hyperbola at the orbital location of the impulsive maneuver.

### *Propagating the spacecraft's trajectory*

This section describes the algorithms used to propagate the spacecraft's trajectory during both the geocentric and heliocentric phases of the mission.

### Geocentric trajectory propagation

This part of the trajectory analysis implements a *special perturbation* technique which numerically integrates the vector system of second-order, nonlinear differential equations of motion of a spacecraft given by

$$\vec{a}(\vec{r}, \vec{v}, t) = \vec{r}''(\vec{r}, \vec{r}', t) = \vec{a}_g(\vec{r}) + \vec{a}_m(\vec{r}, t)$$

where

$t$  = time

$\vec{r}$  = inertial position vector of the satellite

$\vec{v}$  = inertial velocity vector of the satellite

$\vec{a}_g$  = acceleration due to Earth gravity

$\vec{a}_m$  = acceleration due to the Moon

The system of six first-order differential equations subject to Earth gravity is defined by

$$\dot{y}_1 = v_x = y_4$$

$$\dot{y}_2 = v_y = y_5$$

$$\dot{y}_3 = v_z = y_6$$

$$\dot{y}_4 = -\mu \frac{r_x}{r^3} \left\{ 1 + \frac{3 J_2 r_{eq}^2}{2 r^2} \left( 1 - \frac{5 r_z^2}{r^2} \right) \right\}$$

$$\dot{y}_5 = -\mu \frac{r_y}{r^3} \left\{ 1 + \frac{3 J_2 r_{eq}^2}{2 r^2} \left( 1 - \frac{5 r_z^2}{r^2} \right) \right\}$$

$$\dot{y}_6 = -\mu \frac{r_z}{r^3} \left\{ 1 + \frac{3 J_2 r_{eq}^2}{2 r^2} \left( 3 - \frac{5 r_z^2}{r^2} \right) \right\}$$

where  $r = \sqrt{r_x^2 + r_y^2 + r_z^2} = \sqrt{y_1^2 + y_2^2 + y_3^2}$ . In these equations  $\mu$  and  $r_{eq}$  are the gravitational constant and equatorial radius of the Earth, respectively and  $J_2$  is the oblateness gravity coefficient.

The acceleration contribution of the Moon represented by a *point mass* is given by

$$\vec{a}_m(\vec{r}, t) = -\mu_m \left( \frac{\vec{r}_{m-b}}{|\vec{r}_{m-b}|^3} + \frac{\vec{r}_{e-m}}{|\vec{r}_{e-m}|^3} \right)$$

where

$\mu_m$  = gravitational constant of the Moon

$\vec{r}_{m-b}$  = position vector from the Moon to the satellite

$\vec{r}_{e-m}$  = position vector from the Earth to the Moon

### Heliocentric trajectory propagation

The general vector equation for *point-mass* perturbations such as the Moon or planets is given by

$$\ddot{\mathbf{r}} = -\sum_{j=1}^n \mu_j \left[ \frac{\mathbf{d}_j}{d_j^3} + \frac{\mathbf{s}_j}{s_j^3} \right]$$

In this equation,  $\mathbf{s}_j$  is the vector from the primary body to the secondary body  $j$ ,  $\mu_j$  is the gravitational constant of the secondary body and  $\mathbf{d}_j = \mathbf{r} - \mathbf{s}_j$ , where  $\mathbf{r}$  is the position vector of the spacecraft relative to the primary body.

To avoid numerical problems, use is made of Battin's  $F(q)$  function given by

$$F(q_k) = q_k \left[ \frac{3 + 3q_k + q_k^2}{1 + (\sqrt{1 + q_k})^3} \right]$$

where

$$q_k = \frac{\mathbf{r}^T (\mathbf{r} - 2\mathbf{s}_k)}{\mathbf{s}_k^T \mathbf{s}_k}$$

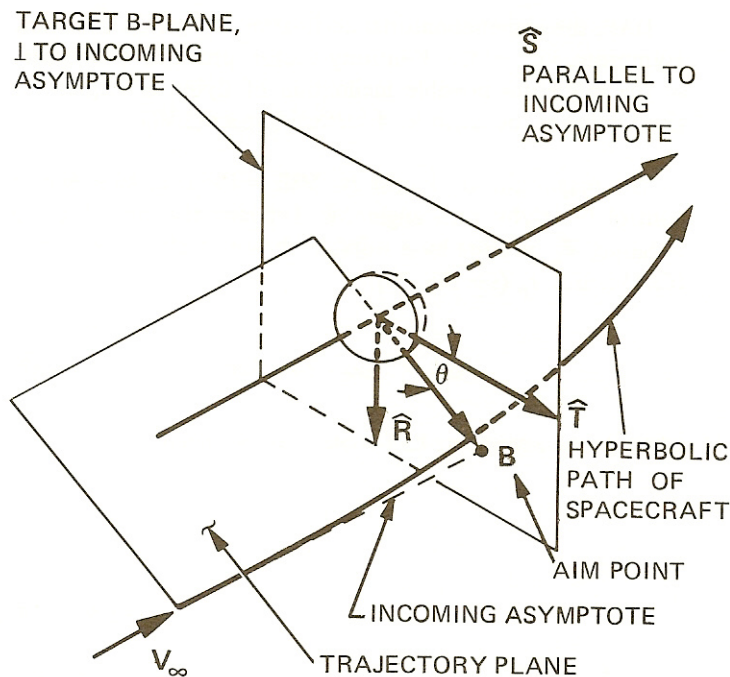
The third-body acceleration can now be expressed as

$$\ddot{\mathbf{r}} = -\sum_{k=1}^n \frac{\mu_k}{d_k^3} [\mathbf{r} + F(q_k) \mathbf{s}_k]$$

In this computer program the heliocentric coordinates of the moon, sun and planets are based on the JPL Development Ephemeris DE421. These coordinates are provided in the Earth mean equator and equinox of J2000 coordinate system (EME2000).

### *B-plane targeting*

The derivation of B-plane coordinates is described in the classic JPL reports, “A Method of Describing Miss Distances for Lunar and Interplanetary Trajectories” and “Some Orbital Elements Useful in Space Trajectory Calculations”, both by William Kizner. The following diagram illustrates the fundamental geometry of the B-plane coordinate system.



The software solves the B-plane targeting problem by minimizing the delta-v vector at the SOI while satisfying two nonlinear *equality constraint* equations. These constraint equations are the differences between components of the *required* B-plane and the B-plane components *predicted* by the software.

Given the user-defined closest approach radius  $r_{ca}$  and orbital inclination  $i$ , and the incoming v-infinity magnitude  $v_\infty$  and the right ascension  $\alpha_\infty$  and declination  $\delta_\infty$  of the incoming asymptote vector at moment of closest approach, the following series of equations can be used to determine the required B-plane target vector:

$$\mathbf{B} \cdot \mathbf{T} = b_i \cos \theta$$

$$\mathbf{B} \cdot \mathbf{R} = b_i \sin \theta$$

where

$$b_i = \sqrt{\frac{2\mu r_{ca}}{v_\infty^2} + r_{ca}^2} = r_{ca} \sqrt{1 + \frac{2\mu}{r_{ca} v_\infty^2}}$$

and

$$\cos \theta = \frac{\cos i}{\cos \delta_\infty}$$

$$\sin \theta = -\sqrt{1 - \cos^2 \theta}$$

$$\sin \delta_\infty = |\hat{\mathbf{s}} \times \hat{\mathbf{z}}| = \sqrt{s_x^2 + s_y^2}$$

$$\hat{\mathbf{z}} = [0 \ 0 \ 1]^T$$

The arrival asymptote unit vector  $\hat{\mathbf{S}}$  is given by

$$\hat{\mathbf{S}} = \begin{Bmatrix} \cos \delta_\infty \cos \alpha_\infty \\ \cos \delta_\infty \sin \alpha_\infty \\ \sin \delta_\infty \end{Bmatrix}$$

where  $\delta_\infty$  and  $\alpha_\infty$  are the declination and right ascension of the asymptote of the incoming hyperbola.

*Important note!!*

This technique only works for aerocentric orbit inclinations that satisfy

$$|i| > |\delta_\infty|$$

If this inequality is not satisfied, the software will print the following error message

```
b-plane targeting error!!
```

```
|inclination| must be > |asymptote declination|
```

It will also display the actual declination of the asymptote and stop. The user should then edit the input file, include a valid orbital inclination and restart the simulation.

The following computational steps summarize the calculation of the *predicted* B-plane vector from a planet-centered position vector  $\mathbf{r}$  and velocity vector  $\mathbf{v}$  at closest approach.

angular momentum vector

$$\mathbf{h} = \mathbf{r} \times \mathbf{v}$$

radius rate

$$\dot{r} = \frac{\mathbf{r} \cdot \mathbf{v}}{|\mathbf{r}|}$$

semiparameter

$$p = \frac{h^2}{\mu}$$

semimajor axis

$$a = \frac{r}{\left(2 - \frac{rv^2}{\mu}\right)}$$

orbital eccentricity

$$e = \sqrt{1 - p/a}$$

true anomaly

$$\cos \theta = \frac{p - r}{er}$$

$$\sin \theta = \frac{\dot{r}h}{e\mu}$$

B-plane magnitude

$$B = \sqrt{p|a|}$$

fundamental vectors

$$\hat{\mathbf{z}} = \frac{r\mathbf{v} - \dot{r}\mathbf{r}}{h}$$

$$\hat{\mathbf{p}} = \cos \theta \hat{\mathbf{r}} - \sin \theta \hat{\mathbf{z}}$$

$$\hat{\mathbf{q}} = \sin \theta \hat{\mathbf{r}} + \cos \theta \hat{\mathbf{z}}$$

S vector

$$\mathbf{S} = -\frac{a}{\sqrt{a^2 + b^2}} \hat{\mathbf{p}} + \frac{b}{\sqrt{a^2 + b^2}} \hat{\mathbf{q}}$$

B vector

$$\mathbf{B} = \frac{b^2}{\sqrt{a^2 + b^2}} \hat{\mathbf{p}} + \frac{ab}{\sqrt{a^2 + b^2}} \hat{\mathbf{q}}$$

T vector

$$\mathbf{T} = \frac{(S_y^2, -S_x^2, 0)^T}{\sqrt{S_x^2 + S_y^2}}$$

R vector

$$\mathbf{R} = \mathbf{S} \times \mathbf{T} = (-S_z T_y, S_z T_x, S_x T_y - S_y T_x)^T$$

### *Targeting to the Mars-centered Periapsis Radius and Orbital Inclination*

For this targeting option, the equality constraints enforced by the SNOPT nonlinear programming algorithm are

$$\begin{aligned} r_p - r_{ca} &= 0 \\ \cos i - \hat{\mathbf{h}}_z &= 0 \end{aligned}$$

where  $r_p$  and  $i$  are the user-defined periapsis radius and orbital inclination, respectively, and  $\hat{\mathbf{h}}_z$  is the z-component of the unit angular momentum vector at closest approach to Mars.

### *Predicting closest approach to Mars*

Closest approach is determined during the numerical integration of the spacecraft's heliocentric equations of motion by finding the time at which the Mars-centered flight path angle is zero. This mission constraint is computed as follows

$$\gamma = \sin^{-1} \left( \frac{\mathbf{r} \cdot \mathbf{v}}{|\mathbf{r} \cdot \mathbf{v}|} \right)$$

where  $\mathbf{r}$  and  $\mathbf{v}$  are the Mars-centered position and velocity vectors, respectively. The numerical method implemented in this computer program consists of the RKF7(8) integrator embedded with a one-dimensional form of Brent's root-finding method.

### *Geocentric-to-areocentric coordinate transformation*

This section describes the transformation of coordinates between the Earth mean equator and equinox of J2000 and areocentric mean equator and IAU node of epoch coordinate systems. This transformation is used to compute the B-plane coordinates at encounter.

A unit vector in the direction of the pole of Mars can be determined from

$$\hat{\mathbf{p}}_{Mars} = \begin{bmatrix} \cos \alpha_p \cos \delta_p \\ \sin \alpha_p \cos \delta_p \\ \sin \delta_p \end{bmatrix}$$

The IAU 2000 right ascension and declination of the pole of Mars in the EME2000 coordinate system are given by the following expressions

$$\alpha_p = 317.68143 - 0.1061T$$

$$\delta_p = 52.88650 - 0.0609T$$

where  $T$  is the time in Julian centuries given by  $T = (JD - 2451545.0)/36525$  and  $JD$  is the TDB Julian Date.

The unit vector in the direction of the *IAU-defined* x-axis is computed from

$$\hat{\mathbf{x}} = \hat{\mathbf{p}}_{J2000} \times \hat{\mathbf{p}}_{Mars}$$

where  $\hat{\mathbf{p}}_{J2000} = [0\ 0\ 1]^T$  is unit vector in the direction of the pole of the J2000 coordinate system.

The unit vector in the y-axis direction of this coordinate system is

$$\hat{\mathbf{y}} = \hat{\mathbf{p}}_{Mars} \times \hat{\mathbf{x}}$$

Finally, the components of the matrix that transforms coordinates from the EME2000 system to the Mars-centered mean equator and IAU node of epoch system are as follows:

$$\mathbf{M} = \begin{bmatrix} \hat{\mathbf{x}} \\ \hat{\mathbf{y}} \\ \hat{\mathbf{p}}_{Mars} \end{bmatrix}$$

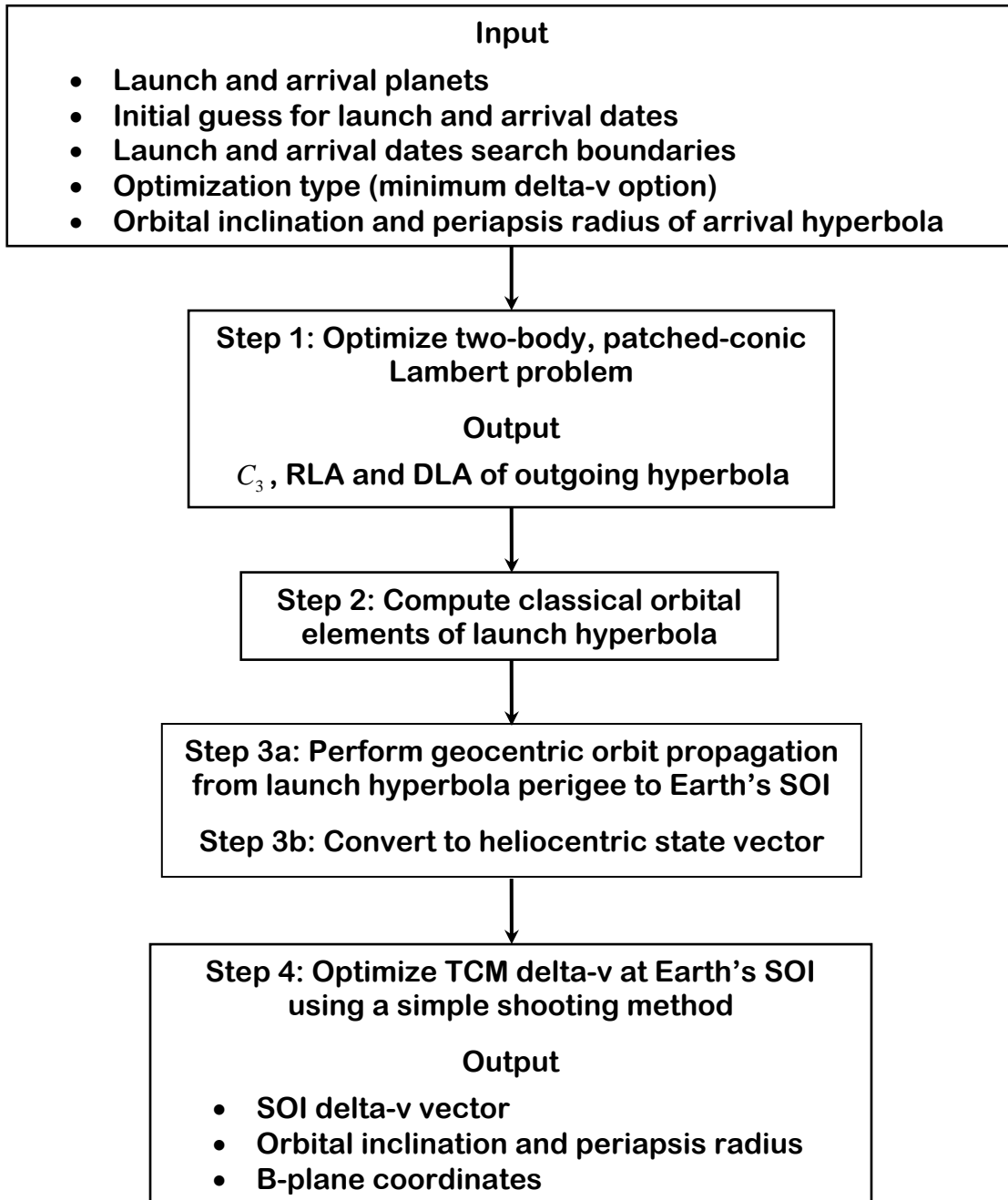
## References and Bibliography

- “Update to Mars Coordinate Frame Definitions”, R. A. Mase, JPL IOM 312.B/015-99, 15 July 1999.
- “JPL Planetary Ephemeris DE410”, E. M. Standish, JPL IOM 312.N-03-009, 24 April 2003.
- “Report of the IAU/IAG Working Group on Cartographic Coordinates and Rotational Elements of the Planets and Satellites: 2000”, *Celestial Mechanics and Dynamical Astronomy*, **82**: 83-110, 2002.
- “IERS Conventions (2003)”, IERS Technical Note 32, November 2003.
- “Planetary Constants and Models”, R. Vaughan, JPL D-12947, December 1995.
- “Preliminary Mars Planetary Constants and Models for Mars Sample Return”, D. Lyons, JPL IOM 312/99.DTL-1, 20 January 1999.
- “Interplanetary Mission Design Handbook, Volume 1, Part 2”, JPL Publication 82-43, September 15, 1983.
- “User’s Guide for SNOPT Version 6, A Fortran Package for Large-Scale Nonlinear Programming”, Philip E. Gill, Walter Murray and Michael A. Saunders, December 2002.

## APPENDIX A

### Computational Chart

This appendix is a summary of the input, output and major computational steps implemented in this computer program.



## APPENDIX B

### Compiling and Running the Software

This appendix describes how to compile and run the e2m computer program. This software was created using version 11.1 of the Intel Visual Fortran and SNOPT, version 7. However, it should compile and run with any “modern” Fortran 77 compiler.

A DOS/Windows version of e2m using Intel Visual Fortran can be created with the following command:

```
ifort e2m.f *.for snopt7.lib /Qsave
```

where snopt7.lib is a compiled library containing the SNOPT subroutines.

An input file created by the user can be run from the command line or a simple batch file with a statement similar to the following:

```
e2m mars03.in
```

If the software is executed without an input file on the command line, the computer program will display the following information screen and file name prompt:

```
*****
*           program e2m           *
*           *                     *
*   Earth-to-Mars trajectory      *
*   design & optimization         *
*           *                     *
*           April 15, 2005        *
*****
```

```
please input the name of the simulation definition file
```

The source code that reads the name of an input file included on the command line is

```
c   if present, use command line argument #1 for input file
    call getarg(1, inputfname$, istatus)
```

The source code that creates the file name input prompt is as follows:

```
c   clear screen

    isys = system("cls")

    if (istatus .eq. -1) then
c   *****
c   input filename not on command line
c   request name of simulation definition input file
c   *****

    print *, ' '
    print *, ' '
```

```

print *, ' *****'
print *, ' *                program e2m                *'
print *, ' *                *                *'
print *, ' *      Earth-to-Mars trajectory      *'
print *, ' *      design & optimization            *'
print *, ' *                *                *'
print *, ' *                April 15, 2005        *'
print *, ' *****'

print *, ' '
print *, ' '
print *, '
&      'please input the name of the simulation definition file'

      read (*, *) inputfname$
end if

```

If your compiler does not accept input from a command line, you will have to modify this source code for your particular Fortran compiler. You may also choose to eliminate the code that accepts a command line input file. Please also note that your compiler may have a different command to clear the screen.

# APPENDIX C

## Contents of the Simulation Summary and CSV Files

This appendix is a brief summary of the information contained in the simulation summary screen displays and the CSV data files produced by the e2m software.

The simulation summary screen display contains the following information:

**transfer time** = total time from the TCM maneuver to closest approach at Mars

**TDB time** = simulation event time on the TDB time scale

**sma (km)** = semimajor axis in kilometers

**eccentricity** = orbital eccentricity (non-dimensional)

**inclination (deg)** = orbital inclination in degrees

**argper (deg)** = argument of periapsis in degrees

**raan (deg)** = right ascension of the ascending node in degrees

**true anomaly (deg)** = true anomaly in degrees

**arglat (deg)** = argument of latitude in degrees. The argument of latitude is the sum of true anomaly and argument of periapsis.

**period (days)** = orbital period in days

**rx (km)** = x-component of the spacecraft's position vector in kilometers

**ry (km)** = y-component of the spacecraft's position vector in kilometers

**rz (km)** = z-component of the spacecraft's position vector in kilometers

**rmag (km)** = scalar magnitude of the spacecraft's position vector in kilometers

**vx (km/sec)** = x-component of the spacecraft's velocity vector in kilometers per second

**vy (km/sec)** = y-component of the spacecraft's velocity vector in kilometers per second

**vz (km/sec)** = z-component of the spacecraft's velocity vector in kilometers per second

**vmag (km/sec)** = scalar magnitude of the spacecraft's velocity vector in kilometers per second

**deltav-x** = x-component of the TLI impulsive velocity vector in meters/second

**deltav-y** = y-component of the TLI impulsive velocity vector in meters/second

**deltav-z** = z-component of the TLI impulsive velocity vector in meters/second

**delta-v** = scalar magnitude of the TLI maneuver in meters/seconds

**b-magnitude** = magnitude of the b-plane vector

**b dot r** = dot product of the b-vector and r-vector

**b dot t** = dot product of the b-vector and t-vector

**theta** = orientation of the b-plane vector in degrees

**v-infinity** = magnitude of incoming v-infinity vector in kilometers/second  
**r-periapsis** = periapsis radius of incoming hyperbola in kilometers  
**decl-asy** = declination of incoming v-infinity vector in degrees  
**rasc-asy** = right ascension of incoming v-infinity vector in degrees

The comma-separated-variable disk file contains the following information:

**time (hrs)** = simulation time since TCM maneuver in hours  
**time (days)** = simulation time since TCM maneuver in days  
**rs2sc-x (au)** = x-component of the spacecraft's heliocentric position vector in astronomical units  
**rs2sc-y (au)** = y-component of the spacecraft's heliocentric position vector in astronomical units  
**rs2sc-z (au)** = z-component of the spacecraft's heliocentric position vector in astronomical units  
**rs2sc-mag (au)** = the spacecraft's heliocentric radius in astronomical units  
**vs2sc-y (km/sec)** = y-component of the spacecraft's heliocentric velocity vector in kilometers per second  
**vs2sc-z (km/sec)** = z-component of the spacecraft's heliocentric position vector in kilometers per second  
**vs2sc-mag (km/sec)** = the spacecraft's heliocentric speed in kilometers per second  
**rm2sc-x (km)** = x-component of the spacecraft's areocentric position vector in kilometers  
**rm2sc-y (km)** = y-component of the spacecraft's areocentric position vector in kilometers  
**rm2sc-z (km)** = z-component of the spacecraft's areocentric position vector in kilometers  
**rm2sc-mag (km)** = the spacecraft's areocentric radius in kilometers  
**vm2sc-y (km/sec)** = y-component of the spacecraft's areocentric velocity vector in kilometers per second  
**vm2sc-z (km/sec)** = z-component of the spacecraft's areocentric position vector in kilometers per second  
**vm2sc-mag (km/sec)** = the spacecraft's areocentric speed in kilometers per second  
**rs2e-x (au)** = x-component of Earth's heliocentric position vector in astronomical units  
**rs2e-y (au)** = y-component of Earth's heliocentric position vector in astronomical units  
**rs2e-z (au)** = z-component of Earth's heliocentric position vector in astronomical units  
**rs2e-mag (au)** = Earth heliocentric radius in astronomical units  
**rs2m-x (au)** = x-component of Mars heliocentric position vector in astronomical units  
**rs2m-y (au)** = y-component of Mars heliocentric position vector in astronomical units  
**rs2m-z (au)** = z-component of Mars heliocentric position vector in astronomical units

**rs2m-mag (au)** = Mars heliocentric radius in astronomical units

**sma-heo (au)** = heliocentric semimajor axis of the spacecraft in astronomical units

**ecc-heo** = heliocentric orbital eccentricity of the spacecraft(non-dimensional)

**incl-heo (deg)** = heliocentric orbital inclination of the spacecraft in degrees

**argper-heo (deg)** = heliocentric argument of perigee of the spacecraft in degrees

**raan-heo (deg)** = heliocentric right ascension of the ascending node of the spacecraft in degrees

**tanom-heo (deg)** = heliocentric true anomaly of the spacecraft in degrees

**sma-areo (km)** = areocentric semimajor axis of the spacecraft in kilometers

**ecc-areo** = areocentric orbital eccentricity of the spacecraft(non-dimensional)

**incl-areo (deg)** = areocentric orbital inclination of the spacecraft in degrees

**argper-areo (deg)** = areocentric argument of perigee of the spacecraft in degrees

**raan-areo (deg)** = areocentric right ascension of the ascending node of the spacecraft in degrees

**tanom-areo (deg)** = areocentric true anomaly of the spacecraft in degrees

The heliocentric coordinates of the spacecraft are with respect to the Earth mean equator and equinox of J2000 (EME2000) coordinate system.. The areocentric coordinates of the spacecraft are with respect to the Mars-centered mean equator and IAU node of epoch coordinate system.

# APPENDIX D

## Fortran Functions and Subroutines

This appendix is a brief summary of the major Fortran functions and subroutines included in the e2m computer program.

- e2m.f** - main executive program
- atan3.for** - four quadrant inverse tangent function
- display.for** - subroutine that displays solution information
- eci2orb.for** - convert eci position and velocity vectors to classical orbital elements subroutine
- findca.for** - find closest approach to Mars subroutine
- findsoi.for** - find time and conditions at the Earth's SOI function
- fpamars.for** - compute Mars-centered flight path angle subroutine
- fpaobj.for** - flight path angle objective function
- funcon1.for** - two-body optimization constraint subroutine
- funcon2.for** - b-plane optimization constraint subroutine
- funobj1.for** - two-body optimization objective function subroutine
- funobj2.for** - b-plane optimization objective function subroutine
- geo\_eqm.for** - geocentric equations of motion subroutine
- hel\_eqm.for** - heliocentric equations of motion subroutine
- lambfunc.for** - solve Lambert's problem subroutine
- launch.for** - computes characteristics of the launch hyperbola
- mm2000.for** - create Earth equator-to-lunar equator transformation matrix
- oeprint.for** - subroutine that displays classical orbital elements
- orb2eci.for** - convert classical orbital elements to position and velocity vectors subroutine
- pleph.for** - main JPL ephemeris subroutine
- readfpn.for** - read and echo floating point number from an input file subroutine
- readint.for** - read and echo an integer from an input file subroutine
- readtext.for** - read and echo text from an input file subroutine
- rkf78.for** - Runge-Fehlberg-Kutta (RK78) numerical integration subroutine
- rkf78cn.for** - evaluate RK78 integration coefficients subroutine

**rmobj.for** - geocentric distance objective function subroutine  
**root.for** - one-dimensional root-finding subroutine  
**rv2bp.for** - state vector to b-plane conversion subroutine  
**sv2000.for** - j2000 lunar and solar ephemeris subroutine  
**twobody2.for** - two-body orbit propagation subroutine  
**utility.for** - number and text manipulation functions and subroutines  
**uvector.for** - unit vector subroutine  
**vcross.for** - vector cross product subroutine  
**vdot.for** - vector dot product subroutine  
**vecmag.for** - vector scalar magnitude function  
**xmod.for** - modulo 2 pi function

## APPENDIX E

### Nonlinear Programming Software

This appendix provides additional information about the two types of optimization performed within the e2m software. It provides Fortran source code that illustrates how the problems are solved with SNOPT, and discusses how the user can adapt the software to use other NLP codes.

As provided, the e2m computer program is written to use the latest version of the SNOPT nonlinear programming software ([www.sbsi-sol-optimize.com/asp/sol\\_product\\_snopt.htm](http://www.sbsi-sol-optimize.com/asp/sol_product_snopt.htm)). However, the software should work with any good multi-dimensional algorithm that can handle bounds on the control variables and equality constraints. Since analytic derivatives for this problem would be difficult to derive, the software should also use some form of numerical differencing for gradient information. Of course, this last requirement would not be necessary if the user chooses to use some form of genetic or evolutionary algorithm.

#### *Two-body optimization*

The part of the code that solves the two-body optimization problem is shown below.

```
c      initial guess
      x(1) = xjdatei1 - xjdate0
      x(2) = xjdatei2 - xjdate0

c      transfer time
      taud = x(2) - x(1)
      tof = taud * 86400.0d0

c      define npopt options
      ip = 0
      is = 0
      call npinit(ip, is, iw, leniw, w, lenw)
      call npseti('derivative level', 0, ip, is, inform, iw, leniw,
&                w, lenw)
      call npsetr('major optimality tolerance', 1.0d-8, ip, is,
&                inform, iw, leniw, w, lenw)

c      control variable lower bounds
      bl(1) = x(1) + dtstep1m
      bl(2) = x(2) + dtstep2m

c      control variable upper bounds
      bu(1) = x(1) + dtstep1p
```

```

        bu(2) = x(2) + dtstep2p
c      number of control variables
        ncv = 2
c      number of linear constraints
        nclin = 0
c      number of nonlinear constraints
        ncnln = 0
        lda = 1
        ldj = 0
        ldr = 2

        call npopt (ncv, nclin, ncnln, lda, ldj, ldr, a, bl, bu,
&                  iptocn, iptofunc, inform, iter, istate, c, cjac,
&                  clamda, f, g, r, x, iw, leniw, w, lenw)

```

### *Numerically integrated optimization*

The source code that solves the TCM optimization, b-plane targeting problem is as follows:

```

c      TCM delta-v vector initial guess (meters/second)
        x(1) = 0.0d0
        x(2) = 0.0d0
        x(3) = 0.0d0
c      TCM delta-v vector lower and upper bounds (meters/second)
        do i = 1, 3
            bl(i) = -50.0d0
            bu(i) = +50.0d0
        end do
c      b-plane components lower and upper bounds
        do i = 4, 5
            bl(i) = 0.0d0
            bu(i) = 0.0d0
        end do
c      define npopt options
        ip = 0
        is = 6
        call npinit(ip, is, iw, leniw, w, lenw)

```

```

    call npseti('derivative level', 0, ip, is, inform, iw, leniw,
&              w, lenw)

    call npsetr('major optimality tolerance', 1.0d-8, ip, is,
&              inform, iw, leniw, w, lenw)

c   number of control variables

    ncv = 3

c   number of linear constraints

    nclin = 0

c   number of nonlinear constraints

    ncnln = 2

    lda = 1

    ldj = 0

    ldr = 2

    call npopt (ncv, nclin, ncnln, lda, ldj, ldr, a, bl, bu,
&              shoot, tcmfunc, inform, iter, istate, c, cjac,
&              clamda, f, g, r, x, iw, leniw, w, lenw)

```

In both optimization problems, we are calling the `npopt` subroutine which is part of the SNOPT software suite. This routine requires a Fortran subroutine that evaluates the objective function (`funobj*.for`) and a second subroutine (`funcon*.for`) that computes the current values of the mission constraints. Other NLP source codes often include both types of computations within a single subroutine.